

微处理机的结构与程序设计

[美] 威廉·F·莱希 著
周邦伟 译 张根度 校

中 国 铁 道 出 版 社

1981年·北京

内 容 提 要

本书系统地介绍了微处理机的结构、程序设计及应用方法。

全书共分十五章。第一章和第二章介绍了微处理机的总体结构和原理；第三章至第九章介绍了微处理机的部件；第十章至第十三章介绍了微处理机程序设计的基本知识和Intel 8008微处理机的硬件结构及机器语言；第十四章通过两个应用实例，介绍了使用微处理机进行程序设计的步骤及方法；第十五章对Intel 8080和M6800微处理机作了扼要的介绍。

本书可供从事微处理机应用的工程技术人员和高等院校师生学习及参考。

William F. Leahy
Microprocessor Architecture
and Programming

John Wiley & Sons, 1977

微处理机的结构与程序设计

〔美〕威廉·F·莱希 著

周邦伟 译

张根度 校

中国铁道出版社出版

责任编辑 郭 宇

封面设计 王毓平

新华书店北京发行所发行

各地新华书店经售

中国铁道出版社印刷厂印

开本：787×1092 $\frac{1}{16}$ 印张：12 插页：1 字数：234千

1981年4月 第1版 1981年4月 第1次印刷

印数：0001—6,000 册 定价：1.30元

译者序

自从七十年代初出现微处理机以来，在十年左右的时间内，用微处理机组成的微型计算机得到了迅速发展。促进微处理机迅速发展的最主要因素是大规模集成电路工艺技术的迅速发展。后者以每年功能增加一倍、价格降低一半的速度发展着。现在已经出现了在性能上相当于、甚至超过小型计算机的微处理机。微处理机的发展也影响了计算机的系统结构：目前LSI存储器几乎完全代替了磁芯存储器；微处理机和其它设备相结合，可以构成功能很强的智能设备；以往共享资源的集中数据系统，现在可以用微处理机组成将资源分散的分布处理系统。

由于微处理机体积小，价格便宜，性能日益提高，能满足多方面应用的需要。因此它扩大了计算机的应用领域，成为计算机应用最活跃的方面之一。

本书对微处理机的结构原理、程序设计及应用方法作了系统的介绍，内容包括中、大规模集成电路工艺及其发展过程，微处理机的硬件组成及其程序设计方法。内容颇为丰富，材料的组织和叙述也比较系统，并且结合实例具体介绍了程序设计的步骤及方法。

本书由于出版的年代较早（1977年），书中还未反映近几年发展起来的微处理机系统。但是，只要掌握了本书介绍的微处理机语言和操作以后，由基本微处理机的总体结构转到目前所发展起来的更为复杂的微处理机系统是不难的。对于初学者来说，基本系统更易入门。

参加翻译工作的还有吴猷定，他完成了第一章至第九章的初稿。翻译过程中，对原书中存在的错误作了校正。

由于水平有限，译文中的错误和不妥之处，请读者批评指正。

译者

一九八〇年五月

序

自从微处理机问世以来，已经在电子界产生了一场技术革命。这场技术革命远远超过了50年代出现的双极型晶体管产生的影响。它不仅影响到技术专业人员（电子工程师、程序员以及科学工作者），而且也开创了一种新的商品——消费者电子设备，打开了小型商业人员的复杂而又“灵巧”的电子系统的市场。

精密的“空间”计算机控制系统，曾经需要化费成千上万美元，并需要一支庞大的工程技术队伍来设计和制造这个系统。现今正在迅速被专用微处理机做的控制器取代，而只要化费很少一部分的费用。新近加入电子行业的人，大量设计和制造了这类系统，他们的企业规模比众所周知的“厨具商店”稍微大一点。过去由于价格昂贵而省去使用微处理机来控制的那些应用领域，现在也因为微处理机的基本价格和维护费用都很便宜而首先考虑选用微处理机做控制器。想使企业“现代化”的商业家或专业人员，必须对出现的新领域卓有远见，否则他们必将被卓有远见的专业人员所替代。

随着微处理机的诞生，传统的模拟和数字工程师的职责也必须作修改。过去曾经是模拟设计工程师设计的范围，现在也正在开始用高精度、高可靠性和低成本的微处理机取代。按照目前趋势，数字技术领域正在从分立的小规模集成元件过渡到使用微处理机系统。整个电子工程界正在经历此种深刻的变化，工程师必须迅速适应，否则就要被淘汰。

简言之，一切领域都在受到微处理机引起的这场革命的影响。可以说，任何一个专业人员，不管他是管理方面的，或者是商业方面的，或者是电子工程方面的，如果不掌握微处理机知识，那末就不可能站在世界技术的前端。

本书打算帮助读者了解从分立的电子技术演变到微处理机用的中规模和大规模集成电路的工业发展过程。对提高微处理机性能和产品销路进行成本核算的商业管理员和 设计 工程师，可以在第一章到第三章中找到有关的参考意见。第四章到第九章描述了微处理机的组成部分，重点放在微处理机存储系统的设计上。因为，用微处理机做控制器时，存储器所用的费用总是占去相当大的部分。

当读者了解了第一章到第九章的微处理机的硬件知识后，可以开始详细学习第十章到第十二章介绍的机器语言程序设计，以及第十三章到第十四章给出的范围广泛的程序设计例子。这些章节都是使用Intel 8008微处理机的机器语言。

本书还包括Intel 8080和Motorola M6800微处理机片程序设计用的材料（见第十五章）。这两种器件是市场上最流行的第二代微处理机片的代表。

威廉·F·莱希

目 录

第一章 导论.....	1
第二章 微处理机——概述.....	3
第一节 计算机的基本部件.....	3
一、运算器.....	3
二、控制器.....	4
三、主存储器.....	7
四、输入、输出部件.....	7
第二节 计算机的操作.....	7
第三章 双极型及金属氧化物半导体技术.....	12
第一节 基本的MOS工艺.....	12
第二节 MOS晶体管的工作原理.....	13
第三节 MOS大规模集成电路的优点.....	14
一、经济因素.....	14
二、性能因素.....	17
第四章 存储器.....	19
第一节 存储系统.....	20
一、二-十进制转换.....	20
二、十-二进制转换.....	22
第二节 存储器的存储方式.....	23
第五章 顺序存取存储器.....	24
第一节 数据格式.....	24
第二节 半导体移位寄存器.....	26
第六章 随机存取存储器.....	28
第一节 存储单元寻址.....	29
第二节 磁芯存储器.....	31
第三节 半导体随机存取存储器.....	33
一、双极型存储单元.....	34
二、MOS存储单元.....	35
三、RAM存储板.....	37
四、存储单元寻址.....	37
第七章 静态的数据和控制存储器.....	45
第一节 经修改的RAM系统.....	45
第二节 半导体ROM系统.....	46
一、掩模编程ROM.....	46

二、可编程ROM	48
三、可擦可编程ROM	50
第三节 可编程逻辑阵列	51
第八章 中央处理机	55
第一节 总线系统	55
第二节 多相时钟和选通信号发生系统	57
第三节 栈指示器系统	60
第四节 运算器	61
第五节 CPU接口	64
第九章 微处理机接口	65
第一节 数字式外围接口	65
一、键盘	65
二、顺序存取设备	69
第二节 模拟接口	71
一、模拟-数字(A/D)转换器(ADC)	71
二、数字-模拟(D/A)转换器(DAC)	73
三、锁相环路转换器	74
第十章 微处理机的程序设计	77
第一节 程序设计语言	77
一、机器语言	77
二、汇编语言	77
三、高级语言	78
第二节 程序设计的流程图方法	79
一、流程图符号	80
二、机器语言的流程图作法	83
三、机器语言的程序设计	84
第十一章 Intel 8008的机器语言	89
第一节 基本指令周期	89
第二节 片内寄存器	91
第三节 Intel 8008指令系统	92
第十二章 十六进制指令码	104
第一节 计算机的数字系统	104
一、八进制数字系统	104
二、十六进制数字系统	104
第二节 十六进制指令码系统	105
第三节 Intel 8008的十六进制指令码矩阵	106
第四节 Intel 8008指令码矩阵的用法	108
第五节 程序纸	109
第十三章 程序设算法入门	112
第一节 程序的循环计数器	112

第二节 算术算法	116
一、加法和减法	116
二、乘法	116
三、除法	119
第三节 控制算法	126
一、不定长的暂停	126
二、固定时间的延时	127
第十四章 系统控制器的硬件和软件	130
第一节 例 1 —— 交通管制器	130
一、时钟功能	131
二、传感器功能	132
三、传感器的请求	133
四、交通管制器的硬件装置	133
五、交通管制的逻辑流程图	135
第二节 例 2 —— 糖果厂的算法	153
第十五章 Intel 8080及M6800简介	164
第一节 Intel 8080微处理机系统	165
一、Intel 8080的指令系统	165
二、Intel 8080的十六进制指令码矩阵	172
第二节 M6800微处理机系统	172
一、M6800的指令系统	172
二、M6800的十六进制指令码矩阵	175

第一章 导 论

微处理机刚开始问世时，在电子工程界引起了各种反应：有的惊奇诧异，有的淡漠，有的兴奋，也有的轻视。在近20年来出现的工程器件中，没有一个象它那样，在开创阶段会在工程界产生如此混杂的感情和受到误解。

作者认为，这种情况主要是器件销售商在向工程界介绍微处理机时所采用的方式造成的。在70年代初期的文献中介绍微处理机，既不在元件、又不在系统的范围之内；那时的微处理机还不能作为解决专用问题的工具，也不适用于通用计算。元件、电路工程师认为微处理机不属于元件，也不属于电路，而是属于系统范围，因此不愿意用它。而系统工程师则因微处理机自身还不能独立完成某个功能，在器件设计师把它做成完整的能够独立完成某个功能的硬件之前，也把它放在一边。而工程程序员又因微处理机与传统的计算机不同，要求它们具有许多非程序设计方面的知识后才能发挥微处理机的功效，所以也避开使用它。总之，按照工程设计职责的传统划分方法，使初期的微处理机成为在工程训练要求上任何人都接受不了的“孤儿”元件。

为了改善这种局面，微处理机销售商以及一些私立教育机构，为工程师们广泛举办了多种讨论性质的专题讲座。这种努力仍然受到缺乏明确的工程训练要求的影响。而此种情况，不但今日存在，预计还要延续一段时间。为了使用微处理机系统，要求设计工程师也是一名计算机程序员，或者要求工程程序员也是一名电路设计师。一般说来，这些都是不切实际的设想。虽然如此，但微处理机将给21世纪的工程问题提供寻找动态系统解的手段，必然会作为设计师和程序员都极其需要的工具而占有重要的地位。

这本讨论微处理机技术的书，打算在工程训练要求的范围之内来介绍微处理机技术，同时指出它可以作为一种工程器件、电路设计的工具。重点放在应用微处理机系统功能的程序设计方面，而程序设计的复杂技巧则最好是留给程序设计的专门训练。

本书尤其着重讨论关于存储器、接口电路和中央处理机（CPU）的程序设计能力在技术上的和经济上的综合方法，这和微处理机系统的设计有关。

应注意，我们并不想使这本讨论微处理机技术的书变成一本充满在当前市场上的微处理机系统和器件的缩影。尽管有很强烈的倾向要我们这么做，但事实证明，这么做是无益的。因为当前在市场上出现的这些器件，其中很多必然最终由于技术的发展而过时，或者由于总体结构不佳而被淘汰。可是没有一个课题的讨论，会象微处理机那样复杂。因此，想完全用一般性的方式来进行讨论而对工程师又是有用的，对微处理机来说是很困难的。由于这一理由，选择最先在市场上出现的8位微处理机英特尔8008（Intel 8008）作为模型器件则既自然又合乎需要。

由于Intel 8008开创了今日为人们所熟知的微处理机形式，因此自然而然地选它作为模型。以它作为研究微处理机总体结构的开始。作为第一个内容，它给出了一个用来确定是否合乎微处理机特征的检验标准。从那以后，产生了大量器件，其中包括第二代微处理机的Intel 8080（见第十五章），还有国家半导体公司（National Semiconductor）、莫托罗拉

(Motorola)、西格奈梯克(Signetics)和其它厂家生产的各种片子。这些后来产生的片子,在指令和存储容量方面大大增强了。它们通常更面向系统,从而很少需要了解微处理机的基本构造。我们相信,只要了解基本的微处理机语言和操作以后,由基本的 Intel 8008微处理机的总体结构转到目前所发展起来的更为复杂的微处理机系统是不难的。

本书所以选择Intel 8008,也是因为现在在廉价出售的电子器件商店里很容易买到Intel 8008片子,而且很便宜(不到20美元一片)。它可以和容易购到的任何存储器片子相配工作,从而学生可以在工厂或家庭实验室中获得动手学习的条件。任何其它学习方法都无法与动手试验的实践学习相比。书本教学是很间接的,要将这种间接的东西通过动手操作电路,将它变成知识。虽然外来的新的微处理机系统正在天天介绍给工程界,但只有少量的系统已成为现成可交付使用的装置。

第二章 微处理机 —— 概述

微处理机是一个缩小到微型尺寸的集成化的计算机。简言之，微处理机就是“单片计算机”。

为了了解微处理机的操作和效能，首先必须了解计算机的基本结构，并把它用于现今已发展的微处理机技术上。在本书中，计算机的定义是：任何一种能自动地对数字或模拟数据进行顺序操作的装置。所说的操作可以是简单的算术运算（加、减、乘、除等）或逻辑判断运算（与、或、等价等）。这些运算用来组合成直接的联机过程控制，或间接的解析分析的复杂计算。

第一节 计算机的基本部件

计算机的基本结构是以四个基本部件为核心的，如图 2—1 所示。这些部件是：

1. 运算部件（ALU），也称运算器；
2. 控制部件，也称控制器；
3. 存储部件，也称存储器，包括程序存储器（PM）和暂存存储器；
4. 输入、输出部件。

为了帮助了解每个部件的操作和它们间的相互关系，对每个部件的基本功能分别进行分析。

一、运 算 器

运算器（ALU）犹如计算机的“肌肉”，它执行逻辑判断和解析计算所必须的基本的算术逻辑运算。可见，ALU 确定了计算机的基本功能。

典型的运算器被设计成能执行多种算术逻辑运算，它包括：

1. 两数相加；
2. 两数相减；
3. 把一个数右移一位；
4. 把一个数左移一位；
5. 两数比较，并判断是否相等；
6. 两数逻辑与；

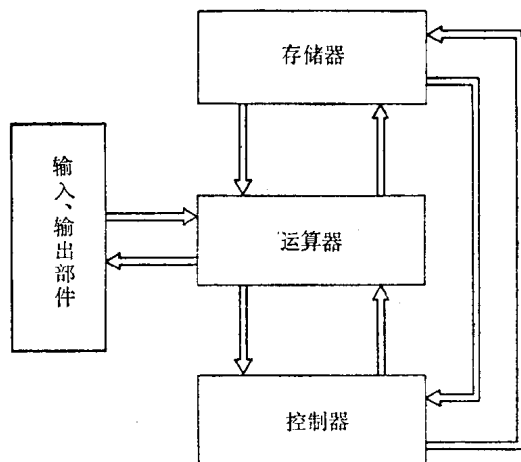


图 2—1 计算机的基本部件

7. 两数逻辑或;
8. 两数逻辑与非;
9. 两数逻辑或非;
10. 两数逻辑异或, 两数逻辑异与

应该强调指出, 这个运算器除了执行简单的加、减运算外, 没有乘、除和其它复杂的算术运算功能。复杂的数学运算和高度复杂的逻辑判断运算都要用计算机的程序来实现。计算机程序(称为算法)确定了如何按顺序组合简单的 ALU 运算, 以完成复杂的数学和逻辑计算。现代计算机就是在这个意义上完成复杂的数学和逻辑运算的。因此, 运算器由于控制器给予的创造能力, 而获得“有限能力”机器人之名。

二、控 制 器

控制器是用来控制计算机的各种运算和指挥所有操作的。简言之, 它起了计算机的大脑中心和神经系统的作用。为了实行这个主要任务, 控制器可再分为四个基本部件, 它们是:

1. 主定时部件;
2. 程序存储器;
3. 栈存储器;
4. 内部程序开关矩阵。

这四个部件之间的相互关系如图 2—2 所示。

(一) 主定时部件

主定时部件主要是用主控定时信号产生和控制计算机的基本时序。只要计算机不被程序(收到一个停机信号)或操作员发出的命令停止运行, 主控定时信号就连续不断地在计算机中重复。图 2—3

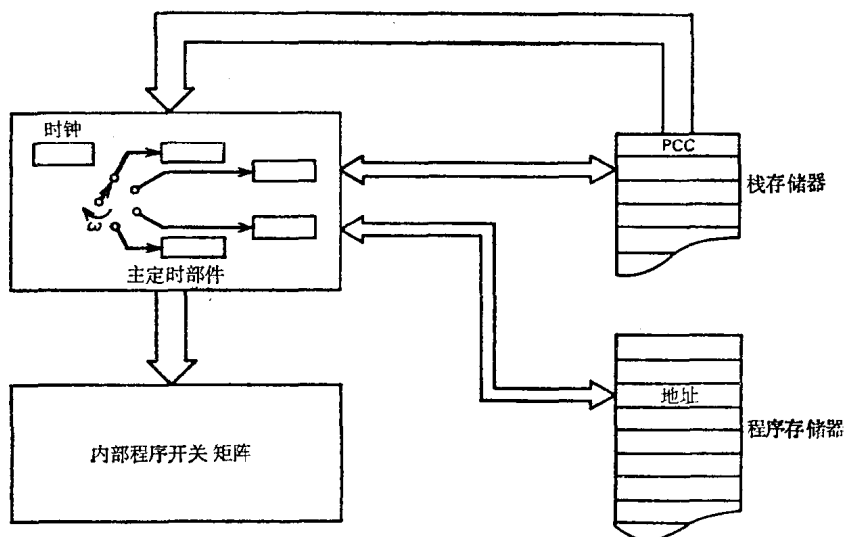
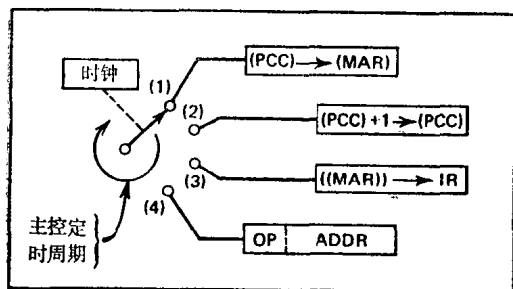


图 2—2 控制器各部件的关系

说明了一个典型的四步指令周期。



PCC——程序计数器;

MAR——存储器地址寄存器;

((MAR))——被寻址的存储单元的内容;

IR——指令寄存器。

图 2—3 四步指令周期

按如下方式组织主控定时信号:

第 1 步 把程序计数器 (PCC) 的值移入存储器地址寄存器 (MAR), 它是当前指令

周期内要执行的指令的存储器地址。

第2步 PCC值加1，为下一个指令周期作准备。按照这种方法，在计算机控制下，整个程序都是按增量方式步进的。通常从0地址开始，每次步进一个计数值。

第3步 把指令送入指令寄存器(指令的地址在第一步中确定)，由它控制计算机中各操作开关的状态；而这些操作开关指挥数据在计算机各部件间传送，确定运算器执行的算术、逻辑运算。

第4步 读指令寄存器内容，适当地接通或断开操作开关，执行指令指定的操作。就本周期内，对数据进行运算并完成指令。

至此，指令周期结束。如果执行的不是停机指令，控制部件将启动另外一个具有同样四步指令周期，但是此刻开始的PCC值比上一个指令周期的PCC值大1。

主控定时信号以及与它相关的四步指令周期的精确定时，是由计算机的时钟控制的。时钟则从主振荡器引出，就象一个恒速旋转的开关那样，按固定重复的时间顺序扫描输出主控定时信号。

(二) 程序存储器

程序存储器是受控制器控制的一个主要部件，它保存了由计算机执行的事件流程（称为计算机的程序）。

在指令周期执行第1步时，PCC确定了控制器将在程序存储器中读的单元地址；一旦该单元读出后，其内容被送入指令寄存器。在指令周期执行第4步时，以数据形式存放在指令器中的操作码建立起计算机的操作开关的状态；这些开关向计算机的其它部件发出适当寄存的命令，并在本步内执行。

程序存储器被称为计算机的大脑，因为它详细记录了计算机将要执行的整个程序。该存储器的内容可以是：

(1) 可变的，在计算机运行前，程序员把一叠新的穿孔卡片读入程序存储器，来改变存储器的内容（通用计算机的情况）；

(2) 永久不变的（专用微处理机的情况），在这种情况下，不对元件作一些改变，就不可能修改程序的任何一步。有关程序设计将在第六、七、十章中说明。

(三) 栈存储器*

和程序计数器、程序存储器密切相关的是栈存储器（也称堆栈存储器）。栈存储器是由许多作堆栈用的存储单元组成，栈顶单元总是被当前程序计数器所占据。在未发生转移情况下，程序计数器在每个指令周期内进行一次计数加1，栈的大小看起来好象是不受限制的。计算机的精华在于它能够按照内部或外部条件来改变程序的执行顺序，栈则被用来记录程序的这种执行顺序的变化。

在典型的程序执行过程中，程序计数器在每个指令周期内加1。用这种方法，一步一步顺序通过程序存储器的每一个存储单元，并按正确的次序读出其内容。然而，如果由于某种判断的结果，程序需要改变这个顺序（转移到不同的程序地址），程序计数器就被修改到新的程序段的起始单元的地址。为了完成对程序计数器的修改，通常把当前程序计数器的值存入（推入）堆栈，而把新的程序计数器值（新的程序段的起始地址）送入栈顶单元。在以后

* 栈，一个特定的存储区，它的一端是固定的，另一端是浮动的。对这个特定的存储区，所有的信息存入和取出都只能在浮动的一端进行，并且符合后进先出的原则。栈也称为堆栈。这个特定的存储区也就称为栈存储器。——译者注

每个指令周期内，程序计数器加1，按新的程序段步进通过程序存储器。而老的程序计数器（处于栈中程序计数器的下面）保留了它的最后一次计数值（地址）。

这种通过推入堆栈来改变程序执行顺序的过程，在栈溢出之前能够继续下去。这样使计算机可以作多重判断。每一次新的程序都是判断的结果，同时栈下推一个单元。这种方法称为嵌套。如果发生栈溢出，最初的程序计数器的值就丢失，并可能引起整个程序出错。

转移到另一段程序，一般说来，程序的这种变化，不是永久不变的，而只是暂时的变化。一旦转移后的程序完成时，就要返回主程序。为了从改变后的程序返回到主程序，在程序中插入一条返回指令。这条指令命令栈存储器在程序计数器（栈顶）中重新建立主程序的程序计数器的值。堆栈上托使程序控制返回到原先的程序计数器的值。当使用的返回指令与程序判断同样多时，堆栈下推和上托的次数相同，从而使原来的程序计数器回到栈顶单元。图2—4(a)说明了堆栈下推的方法；而图2—4(b)说明了堆栈上托的情况。

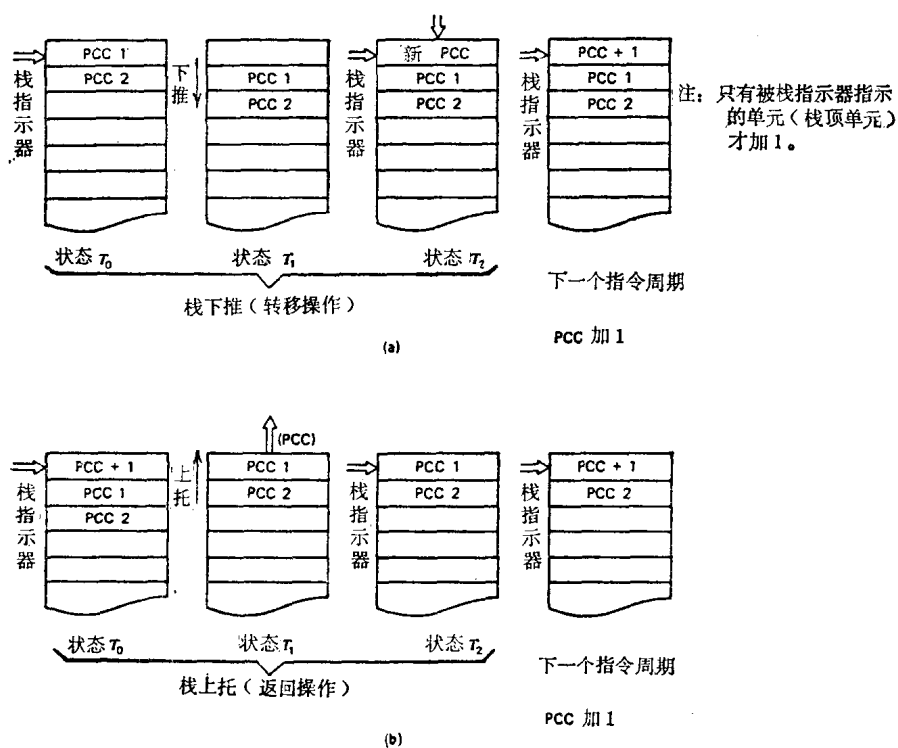


图2—4 栈存储器的下推和上托

(四) 内部程序开关矩阵

内部程序开关矩阵（操作开关）是计算机控制数据交往的控制部分。控制器的其它部分则操纵内部开关矩阵的开关顺序接通或断开，以执行和计算机有关的各种操作。实际的数据也是通过开关矩阵的控制而在计算机内部和外部各个元件中传送的。

除了受到ALU功能限制外，计算机的处理能力与矩阵的复杂程度有关，诸如把数据从累加器传送到其它暂存储器或主存储器，或者反过来把数据从暂存储器或主存储器传送到累加器的数据传送功能。开关矩阵也对输入、输出部件进行控制，控制计算机的内部操作和外部环境之间的数据传送。

三、主 存 储 器

计算机内部结构中最大的部件之一是存储器，存储器一般有两种功能：

- (1) 用来保存控制器操作指令的程序存储器；
- (2) 存储数据。

程序存储器，根据计算机的设计，可以是读、写存储器，也可以是只读存储器。在微处理机的专门应用中，程序存储器通常是由永久性的只读存储器组成。用于数据存储的那部分存储器总是读、写存储器，它可以存储：(a)要用的输入数据；(b)中间计算结果；(c)向外部设备输出的最终结果。

在大多数计算机中，存储器占了计算机系统电路的很大一部分。这是因为运算器(ALU)和控制器比较单一，而程序存储器以及它的存储操作和将结果重新存入数据存储器是比较多样和分散的。因此，虽然从整体上来看存储器的用法是简单的，但它是计算机中主要的硬设备。

四、输入、输出部件

计算机的输入、输出部件是人和计算机，或者计算机和其它机器之间的连接器。一般说来，当提起计算机的输入部件时，就会联想到电传打字机和数字键盘。而当提起输出部件时，就会联想到打印机和显示屏。这些输入、输出部件是连接人和计算机的很好的连接器。但它们不是用来连接计算机和其它机器的连接器。计算机和其它机器连接的输入部件有：温度传感器、压力传感器、光电转换器、机电位置转换器和其它物理参量传感器等。当然，这些传感器所测得的参数必须转换成和计算机相对应的数字格式。有关计算机和其它机器连接的输出部件有：电动机、线圈和机电阀门等。

在上述各种情况中，输入、输出部件实际上是和计算机的输入或输出寄存器相连接的外围设备。这些寄存器用来构成计算机的输入和输出，因为任何一种提供合适数字接口的外围设备，都要有这种输入、输出部件的接口寄存器时才能使用。

输入、输出寄存器是能够把外围设备送来的数据（输入时），或把计算机送来的数据（输出时）暂存起来的寄存器。在使用前，它保存这些数据。一旦这些数据被计算机（输入时）或被外围设备（输出时）接收，寄存器就解锁并准备接收下一组数据。

第二节 计算机的操作

为了说明图2—5所示结构的典型计算机的操作，对执行两数相加的两个系统进行比较。第一个系统是人—袖珍计算器系统，第二个系统是全自动的微处理机系统。

在人—计算器系统中，存储系统（程序存储器和暂存存储器）是一张纸。程序存储器是写在纸上的指令表。它指导人在计算器上进行适当的操作，来求得问题的解。对于在 $x = 6 + 8$ 中求解 x 的问题，可按表2—1的过程执行人—计算器的程序。

这个程序需要16步7个指令周期执行完（注意，一个指令周期完成指令中的基本命令所需要的时间，是随命令所需的持续时间而变化的）。指令周期是机器执行时间的度量单位，

表 2—1

程 序	系 统 操 作
1. 清除计算器	1 个指令周期 { 1. 读: 操作员读程序的第一条指令。 2. 执行: 操作员按计算器的“清除键”。
2. 把数 A 送入计算器	1 个指令周期 { 3. 读: 操作员读程序的第二条指令。 4. 取: 操作员往存储器(纸)中看, 寻找数 A 的值——(6)。 5. 执行: 操作员按计算器的数字键“6”。
3. 按加法功能键 (+)	1 个指令周期 { 6. 读: 操作员读程序的第三条指令。 7. 执行: 操作员按加法功能键。
4. 把数 B 送入计算器	1 个指令周期 { 8. 读: 操作员读程序的第四条指令。 9. 取: 操作员往存储器(纸)中看, 寻找数 B 的值——(8)。 10. 执行: 操作员按计算器的数字键“8”。
5. 按等于功能键 (=)	1 个指令周期 { 11. 读: 操作员读程序的第五条指令。 12. 执行: 操作员按等于功能键(=)。
6. 读和记录答案	1 个指令周期 { 13. 读: 操作员读程序的第六条指令。 14. 执行: 操作员读答案(14)并把它记录在纸上。
7. 停止	1 个指令周期 { 15. 读: 操作员读程序的第七条指令。 16. 执行: 操作员停止工作。

在上例中, 一个指令周期至少要有读和执行两步组成。然而, 很可能有的指令周期要多于两步。

当不用人-计算器系统而用计算机系统来解这个问题时, 必须扩展原来的程序(这是因为计算机没有独立思考的能力, 所以全部操作必须是唯一确定的), 使得控制器能指挥整个计算机系统的操作顺序。程序放在程序存储器里, 计算机在某一时刻执行的指令地址, 保存在程序计数器中。此外, 为了输入数据和输出答案, 还提供一个输入、输出外围设备(电传机), 来和计算机的运算器相接。

图 2—6 描述了执行加法运算时所需要的计算机的四个主要部件之间最必要的逻辑互联。图中所画的开关是受正要执行的指令控制, 并且是内部开关矩阵的一部分。

下面使用大型计算机和微型计算机均适用的简化符号来描写计算机的操作。寄存器用单括号括起表示“寄存器的内容”。两个表达式之间的箭头表示数据在两个寄存器之间按箭头方向进行传送。双括号表示存储单元的内容, 它的地址由内层括号的内容指定。表 2—2 说明了这些简化符号。计算机系统的机器语言程序可按表 2—3 给定的方式书写。

表 2—4 给出了执行这个程序的机器语言的指令序列。程序计数器的起始值为 0000, 处理机调出存在程序存储器 0000 单元中的指令, 对操作(输入数据)进行译码, 并且执行指令, 向输入部件发出控制信号。在完成 0000 单元的指令后, 程序计数器已经加 1, 调出 0001 单元

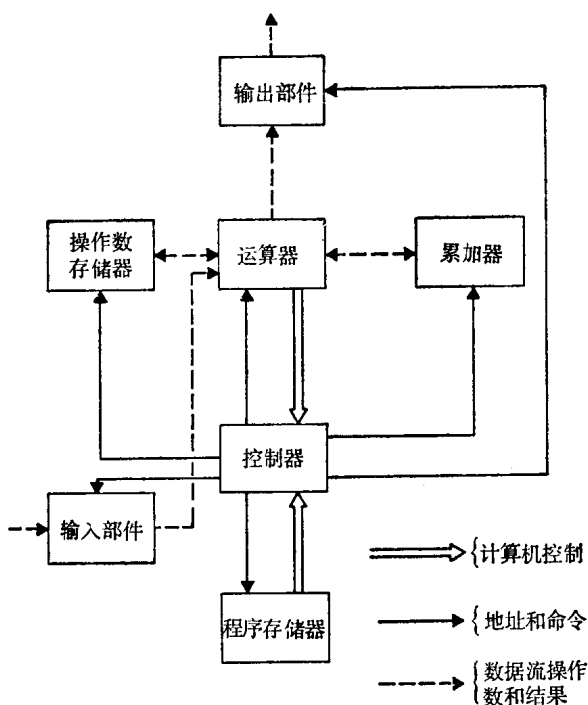


图 2—5 计算机方框图

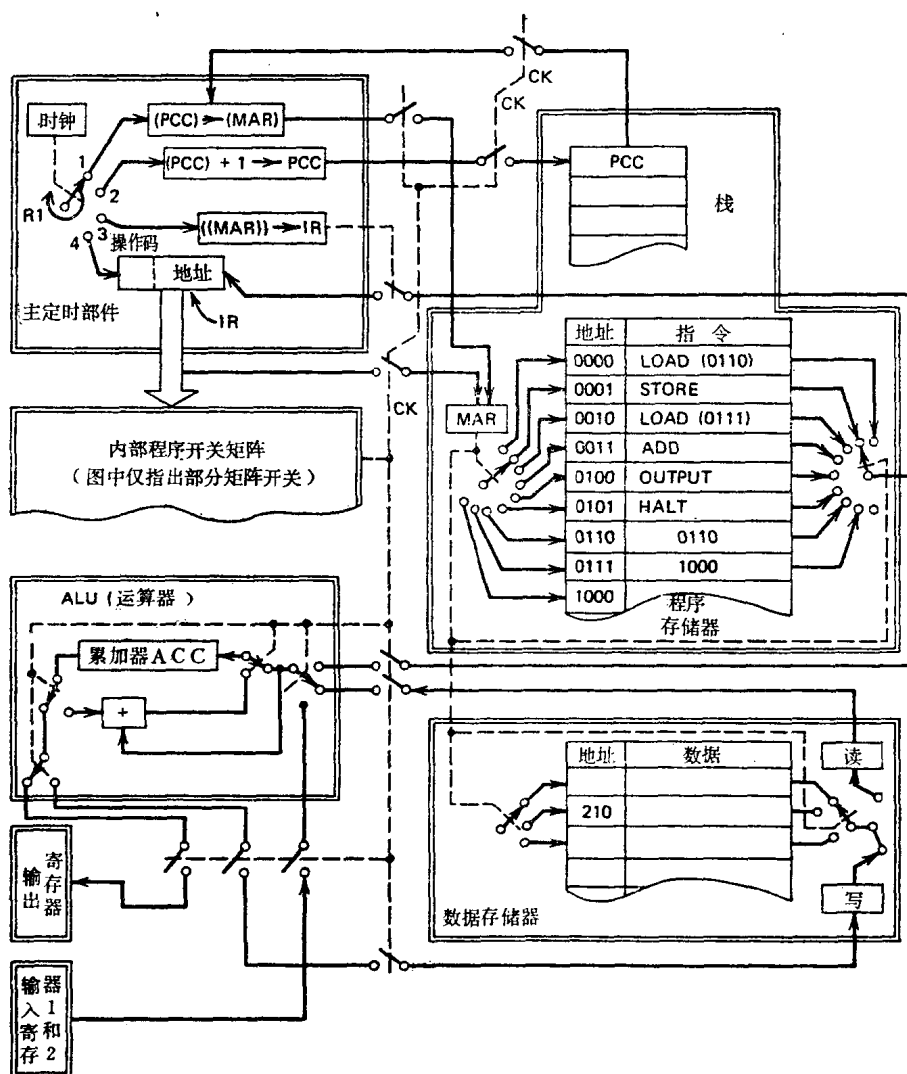


图 2-6 求解 $6 + 8 = ?$ 的系统互联

计算机的简化记号

表 2-2

记号	意义
(PCC)	程序计数器的内容。
(PCC) → MAR	把程序计数器的内容送到存储器的地址寄存器 MAR。
((MAR)) → IR	把存储器地址寄存器指出的单元内容送到指令寄存器 IR。

表 2-3

指令单元地址——PCC	指令	系统操作
0000	把第一个数据输入到暂存器——累加器 A。	在输入终端上, 把数 (6) 送到 CPU 的累加器。累加器通常作输入和输出数据的暂存器。
0001	把累加器数据送到存储单元 210。	把数 (6) 从累加器传送到存储单元 210。
0002	把第二个数据输入到暂存器(累加器)。	在输入终端上, 把数 (8) 送到 CPU 的累加器。
0003	把存储单元 210 的内容加到累加器中。	把存储单元 210 中的数 (6) 和累加器中的当前数 (8) 相加, 产生结果 (14) 存在累加器中。
0004	累加器输出。	把累加器中的数 (14) 在输出部件上输出。
0005	停机。	计算机系统停止工作。

机 器 语 言 的 指 令 序 列

表 2—4

指令周期 序号 N_i	开 关 1	用简化记号的描述	说 明
1.	A B C D	START	操作员按下启动按钮, 接通时钟, 顺序打入两个数 6 和 8。
		$(PCC) \rightarrow MAR$	把 PCC 的内容装入存储器地址寄存器 MAR, 指向存储单元 0000。
		$(PCC) + 1 \rightarrow PCC$	PCC 变为 0001。
		$((MAR)) \rightarrow IR$	第一条指令 (单元 0000) 进入指令寄存器 IR。
2.	A B C D	$((Input\ 1)) \rightarrow ACC$	执行指令, 把第一个数据 (Input 1) 输入累加器 ACC。
		$(PCC) \rightarrow MAR$	MAR 指向存储单元 0001。
		$(PCC) + 1 \rightarrow PCC$	PCC 变为 0010。
		$((MAR)) \rightarrow IR$	第二条指令 (单元 0001) 进入指令寄存器 IR。
3.	A B C D	$((ACC)) \rightarrow 210$	执行指令, 把数据从累加器送到单元 210。
		$(PCC) \rightarrow MAR$	MAR 指向存储单元 0010。
		$(PCC) + 1 \rightarrow PCC$	PCC 变成 0011。
		$((MAR)) \rightarrow IR$	第三条指令 (单元 0010) 进入指令寄存器 IR。
4.	A B C D	$((Input\ 2)) \rightarrow ACC$	执行指令, 把第二个数据 (Input 2) 输入累加器。
		$(PCC) \rightarrow MAR$	MAR 指向存储单元 0011。
		$(PCC) + 1 \rightarrow PCC$	PCC 变成 0100。
		$((MAR)) \rightarrow IR$	第四条指令 (单元 0011) 进入指令寄存器 IR。
5.	A B C D	$((ACC)) + ((210)) \rightarrow ACC$	执行指令, 累加器中数据和存储单元 210 中数据相加, 并把和 ($6 + 8 = 14$) 送入累加器。
		$(PCC) \rightarrow MAR$	MAR 指向存储单元 0100。
		$(PCC) + 1 \rightarrow PCC$	PCC 变为 0101。
		$((MAR)) \rightarrow IR$	第五条指令 (单元 0100) 进入指令寄存器 IR。
6.	A B C D	$((ACC)) \rightarrow Output$	执行指令, 把累加器中数据 (14) 送到输出部件输出。
		$(PCC) \rightarrow MAR$	MAR 指向存储单元 0101。
		$(PCC) + 1 \rightarrow PCC$	PCC 变为 0110。
		$((MAR)) \rightarrow IR$	第六条指令 (单元 0101) 进入指令寄存器 IR。
		(HALT)	执行指令, 断开时钟, 使计算机停止工作。

中的指令进行译码, 发出适当的命令信号, 执行指定的指令。这个过程继续到程序计数器到达 0005 号指令, 计算机停止工作。应该指出, 接通计算机系统电源时, 将强迫程序计数器指向第一条指令单元, 在本例为 0000 单元。常把这种初始化称为计算机的总清。

除了人工向系统提供二个相加数外, 完成两数相加任务, 在人和系统之间并不发生关系。计算机完全按存储器内部的程序进行工作, 在执行过程中它是系统的不变部分。

所有微处理机 (以及所有大型计算机), 都按上述过程进行工作。在集成电路片中的各个部件的位置, 可以随不同的器件制造厂而有变化, 但是总的方框图都与图 2-7 一样。计算机每个部件的详细工作在以后几章讲述。